Honey Bee Based Vehicular Traffic Optimization and Management

Conference Paper *in* Advances in Intelligent Systems and Computing · December 2012
DOI: 10.1007/978-81-322-1041-2_39

3 authors:

Prasun Ghosal
Indian Institute of Engineering Science and T…
98 PUBLICATIONS 246 CITATIONS

Arijit Chakraborty
Indian Institute of Engineering Science and T…
14 PUBLICATIONS 15 CITATIONS

Sabyasachee Banerjee
Heritage Institute of Technology
14 PUBLICATIONS 15 CITATIONS

Honey Bee Based Vehicular Traffic Optimization and Management

Prasun Ghosal[1], Arijit Chakraborty[2], Sabyasachee Banerjee[3]

[1] Bengal Engineering and Science University, Shibpur, Howrah, WB, India

[2] Heritage Institute of Technology, Kolkata, WB, India

{prasun@ieee.org; arijitchakraborty.besu@gmail.com; sabyasasachee.banerjee@gmail.com}

Abstract. Traffic densities in highly populated areas are more prone to various types of congestion problems. Due to the highly dynamic and random character of congestion forming and dissolving, no static and pre deterministic approaches like shortest path first (SPF) etc. can be applied to car navigators. Sensors are adequate here. Keeping view in all the above mentioned factors, our contributions in this paper include the development of a novel Bio Inspired algorithm on multiple layers to solve this optimization problem, where, car routing is handled through algorithms inspired by nature [Honeybee behavior]. The experimental results obtained from the implementation of the proposed algorithm are quite encouraging.

Keywords: Vehicular traffic management; Honey bee; Traffic optimization

1 Introduction

Traffic densities in highly populated areas are more prone to congestion problems, to some extent. Due to highly dynamic and random character of congestion forming and dissolving, no static and pre deterministic approaches like shortest path first (SPF) etc. can be applied to car navigators.

In this paper we tried to emphasize on the progress of a highly adaptive and innovative algorithm to deal with this problem. We got inspired primarily by the ideas of Swarm Intelligence technique, as the same is being applied rationally to address similar kinds of problems since decades back and also that have been detected in the honeybee communication. As a major development towards this field we present an idea based on honeybee based self-organizing vehicle routing algorithm termed as *honey jam* (ideally aimed at traffic congestion).

2 Historical Background

Honey bees live in a colony and play two types of functional roles. In one type, they discover new food sources and are termed as *scouts*, and the second one being *foragers*, that transport nectar from an already discovered flower site by following the dances of other scouts or foragers. Foragers use a special kind of mechanism called waggle dance to specify information about the quality, direction and distance to a distant food source. The intensity of the dance (reflecting the quality of the food source) determines the number of additional foragers required to be recruited to exploit the source. These foragers fly in the rough direction of the food source.

Once they have arrived at the approximate location, the foragers use their senses to precisely find their destination. These recruited foragers arrive in greater numbers at more profitable food sources because the dances for richer sources are more conspicuous and hence likely to be encountered by more number of unemployed foragers.

3 Previous Works

The paper proposed by Jake Kononov, Barbara Bailey, and Bryan K. Allery [1], first explores the relationship between safety and congestion and then examines the relationship between safety and the number of lanes on urban freeways. The relationship between safety and congestion on urban free-ways was explored with the use of safety performance functions [SPF] calibrated for multi-lane free-ways in Colorado, California, Texas. The Focus of most SPF modeling efforts to date has been on the statistical technique and the underlying probability distributions. The modelling process was informed by the consideration of the traffic operations parameters described by the Highway Capacity Manual [1].

In 2006, H Ludvigsen et al., has published Differentiated speed limits allowing higher speed at certain road sections whilst maintaining the safety standards are presently being applied in Denmark [2]. The typical odds that higher speed limits will increase the number of accidents must thus be beaten by the project [2].

In another important work, C.J. Messer et al. [3] presented a new critical lane analysis as a guide for designing signalized intersections to serve rush-hour traffic demands. Physical design and signalization alternatives are identified, and methods for evaluation are provided. The procedures used to convert traffic volume data for the design year into equivalent turning movement volumes are described, and all volumes are then converted into equivalent through auto-mobile volumes. The critical lane analysis technique is applied to the proposed design and signalization plan. The resulting sum of the critical lane volumes is then checked against established maximum values for each level of service (A, B, C, D, E) to determine the acceptability of the design. In this work, the authors have provided guidelines, a sample problem, and operation performance characteristics to assist the engineer in determining satisfactory design alternatives for an intersection [3].

There is one more design called Design of a Speed Optimization Technique in an Unplanned Traffic (DSOTU) [4] finding methods in other literature are a family of optimization algorithms, which incorporate level of traffic services in the algorithms. There are two major issues, in the first part; we have analyzed the major issues residing in the latest practice of the accidental lane; and, in the last part, we have discussed the possible applications of this new technique and new algorithm [4]. Other works in this area are also reported in [5-8].

4 Proposed Work

In our previous work we specially concentrated on maximum speed utilization of any vehicle as well as planning lanes for an unplanned traffic, but in this work we are also considering the speed of each lanes and their speed difference. As too much speed difference drives vehicles to be biased to only one lane though other lanes are not utilized properly. So our algorithm ensures maximum utilization of the lanes present in traffic without affecting the optimum speed of the vehicle too much, because vehicles can transit after the lanes threshold value is reached. But as a trade of, this eventually increases the number of transitions required to give a vehicle its optimum high speed.

4.1 Assumption

To implement this algorithm as a simulation of the real life scenario under consideration, certain assumptions are made without loss of generality of the problem. During the execution of the algorithm it is assumed that there will be no change in the current speed of the vehicle, if accidentally any vehicle's speed becomes '0' then totally discard the vehicle from the corresponding lane. Our algorithm runs periodically and continuously tries to optimize the speed of the lanes by reducing the speed difference of present lanes, but to achieve this we might have to increase the number of transitions of vehicles entered in to the lanes.

4.2 Description of the proposed algorithm

The primary sections of the proposed algorithm and their major functionalities are described below.

Step 1. During this step, inputs are taken from sensors, e.g. current speeds of vehicles, arrival time etc., and numbers of vehicles are counted input by the sensor, and numbers of lanes present in the traffic with their corresponding threshold values are input, too.

Step 2. In this step, lanes are assigned to different vehicles having different current speeds. The way is first fill up the first lane up to its threshold value then when the first lane's threshold value is filled up the vehicles which are allocated to the first lane is moved to the next lane until its population reaches to the threshold value and the population of the first lane get decremented as vehicles moved from the first lane to next lanes. Then the first lane is also get populated with remaining vehicle's speed simultaneously. This process is continued until all the lanes got filled up to its threshold value.

Step 3. Categorizing them depending on their assigned lane.

Step 4. This step finds the lane for remaining number of vehicles, where, the difference between the vehicle's current speed and lane's speed buffer's average speed is minimum and takes the vehicle to the lane, categorizes it same as the lane's other vehicles, increases the population of the lane, and stores the vehicle's current speed in the speed buffer of the lane.

Step 5. This step is used for checking total numbers of transitions, i.e. at which point of the lane and from which lane to where the transition will occur for all vehicles, thereby calculating the average speed of the lanes also.

Table 1. Table of Notations used in algorithm HoneyJam

Notations used	Meaning
V_i	Speed of the ith vehicle
Li	Lane of the ith vehicle
type(i)	Type of the vehicle
T	Arrival time difference between a high and low speed vehicles
t1	Time interval to overtake a vehicles at lower speed
D	Distance covered by low speed Vehicle
d1	Distance covered by high speeding Vehicle
Bn	Buffer of Lane n or population of Lane n
Count	Total no. Vehicle in traffic
t_Count	Total no. of transitions while assigning vehicles to a particular lane up to it's threshold value
trans_l	Total number Of transitions
L_th(i)	Threshold value of lane i
L_V a(i)	Speed of the vehicle a present in the lane i
L_avg(i)	Average speed of the lane i
Lane	Number of lanes
X	Marking after all the lanes filled up by its threshold value

4.3 Pseudo Code of Algorithm HoneyJam

INPUT: Vehicle's name, current speed, arrival time, number of lanes, threshold value of each lane.
OUTPUT: Total number of transitions and the average speed of the each lane present in the traffic.

```
Step 1.1: Set count=1; /*used to count the number of vehicles*/
Step 1.2: get_input ()/*Enter the inputs when speed of the vehicle is non-zero. */
Step 1.3: Continue Step 1.1 until sensor stops to give feedback.
Step 2: for b=1 to Lane
            Set x=x+L_th(b);
            Set Bb=0 and i=1;
             While (Bb! =L_th (b))
             if(b==0)
                         Enter Vi into b lane's speed buffer,
                         Bb++;
                         i++;
             else
                         Enter Vi into 1st lane's speed buffer and transfer 1st lane's
populated vehicle to next                            unpopulated lane.
                         t_Count= t_Count+(b-1);
                         Bb++;
                         i++;
             End Loop
           End Loop
          return x;
Step 3:  for b=1 to Lane
             for a=1 to L_th(b)
                         for c=1 to Count
                                     if(Vc==L_V a(b))
                                                 type(c)=b
                         End Loop
             End Loop
            End Loop
Step 4:  for a=x to Count
             set min=|Va-L_avg(1) |;
             set buf=1;
             set type(a)=1;
                         for b=1 to Lane
                                     d=|Va-L_avg(b)|;/*Taking only magnitude of the
difference*/
                                     if(d==0)
                                                 set type(a)=b;
                                                 set buf=b;
```

```
                        break;
                if(d<min)
                        set type(a)=b
                        set buf=b;
        End Loop
        update Bbuf++;
        update L_V Bbuf(buf);
    End Loop
Step 5: for 1≤i≤Count
    for 2≤j≤Count
        If type(i)= type(j) and Vi<Vj and (i) vehicle’s arrival time≤ (j)
vehicle’s arrival time
        Set t=(j) vehicle’s arrival time - (i) vehicle’s arrival time
        Set t1=0
        Begin loop
            Set t1=t1+1
            Set d=Vi*(t+t1)
            Set d1=Vj*t1
            If d1≤d set trans_l = trans_l+1;
        End loop
        trans_l= trans_l+t_Count;
    End loop
  End loop
  For 1≤m≤Lane
  Calculate each lane’s average speed from its speed buffer.
Step 6: Return Number of transitions required= trans_l
Step 7: End.
```

5 Results and Analysis

5.1 Analysis of Algorithm HoneyJam

- The algorithm stated above is implemented on a planned traffic area where number of lanes and their population's threshold value is known beforehand.
- The objective will follow linear queue as long as speed/value/cost of proceeding to greater than the immediate next.
- Transitions/Crossovers are calculated and appropriate data structures are used in order to maintain the uniformity.

- We assume that the lanes are narrow enough to limit the bi-directional approach.

- We tried to implement logic in the algorithm *HoneyJam* in order to maintain optimized speed for each lane by reducing the average speed difference amongst lanes.

- The algorithm also calculates the transition points if speed/value/cost of a vehicle whenever found unable to maintain the normal movement and failed to transit in all possible calculated lanes.

- Transition points are recorded with their positions and numbers using appropriate data structures to maintain the record.

5.2 Time Complexity Analysis

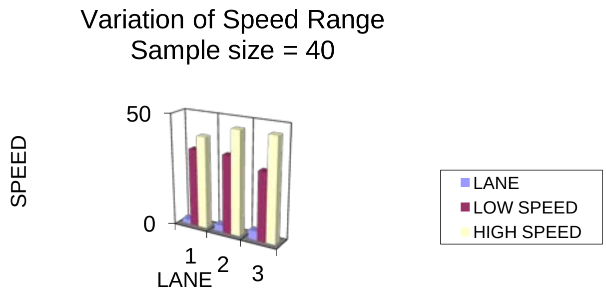
Time complexity of the proposed algorithm and its subsections has been analyzed in the following Table 2.

Table 2. Time complexity analysis

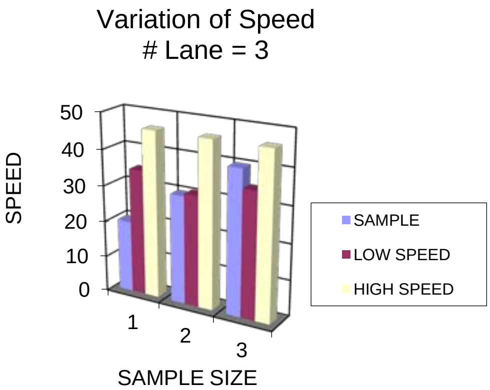
Section	Time complexity	Final Time complexity
Step 1	O(Count)	O ((Count-1)*(Count-1)) Since Count>>Lane and L_th(b)
Step 2	O(Lane*L_th(b))	
Step 3	O(Lane*L_th(b)*Count)	
Step 4	O(Count-(Lane*L_th(b))*Lane)	
Step 5	O((count-1)* (count-1))	

5.3 Graphical Analysis

The proposed algorithm has been implemented using C++ under GNU GCC compiler environment running Linux operating system with an Intel 3 GHz chip and 1 GB of physical memory. Experimental datasets have been plotted in bar chart form to study the variations. Some sample bar charts are shown in the following Figure 1. In the graphical analysis we can easily find that the speed difference between the lanes are far much decreased in the bar chart (Figure 1a) we have fixed the sample size and varied the number of lanes present in the traffic and in next bar chart (Figure 1b) we have fixed the number of lanes present in the traffic and varied the number of samples.



(a)



(b)

Fig. 1 Graphical plot of experimental results

6 Conclusions and Future Scopes

The article presented through this paper mainly emphasize on optimal usage of lanes using threshold information as knowledge base, but at the cost of transitions, because in real life scenario transitions may be too high, hence our future effort will be certainly in this direction.

In this article amount of time taken to transit between lanes has been considered cannot be ignored. The cumulative sum of transition time between lanes in real world problem contributed much in optimality of the proposed solution.

Bio inspired algorithms (like swarm intelligence) has been used with population information as knowledge base in our previous works, but partial modification of the stated concept taking threshold level information of the respective lanes will certainly be taken into consideration but implementation and formulation of algorithms along with optimality in transition, there by optimizing various aspects of traffic movement in real world will be considered in our future effort.

References

[1] Jake Kononov, Barbara Bailey, and Bryan K. Allery, *"The relationship between safety and congestion"*, Journal of the Transportation Research Board, No. 2083.

[2] *"Differentiated speed limits"*, European Transport Conference Differentiated speed limits, 2007.

[3] C.J. Messer and D.B. Fambro, *"Critical lane analysis for intersection design"*, Transportation Research Record No. 644; 1977, pp, 26-35.

[4] Prasun Ghosal, Arijit Chakraborty, Amitava Das, Tai-Hoon Kim, Debnath Bhattacharyya, *"Design of Non-accidental Lane"*, In Advances in Computational Intelligence, Man-Machine Systems and Cybernetics, pp. 188-192, WSEAS Press, 2010.

[5] Prasun Ghosal, Arijit Chakraborty, Sabyasachee Banerjee, *"Design of Efficient Knowledge Based Speed Optimization Algorithm in Unplanned Traffic"*, IUP Journal of Computer Sciences.

[6] Prasun Ghosal, Arijit Chakraborty, and Sabyasachee Banerjee, *"Particle Swarm Optimization of Speed in Unplanned Lane Traffic"*, International Journal of Artificial Intelligence & Applications (IJAIA), Vol.3, No.4, July 2012, pp. 51-63. DOI : 10.5121/ijaia.2012.3404

[7] Prasun Ghosal, Arijit Chakraborty, Sabyasachee Banerjee, and Satabdi Barman, *"Speed Optimization in Unplanned Traffic Using Bio-inspired Computing And Population Knowledge Base"*, Computer Science & Engineering: An International Journal (CSEIJ), Vol. 2, No. 3, June 2012, pp. 79-97. DOI : 10.5121/cseij.2012.2307

[8] Prasun Ghosal, Arijit Chakraborty, and Sabyasachee Banerjee, "*Computational Optimization of Speed in an Unplanned Lane Traffic*", IEM International Journal of Management & Technology (IEMITMT) [ISSN: 2296-6611], pp. 160-163.

[9] P Ghosal, A Chakraborty, S Banerjee, "Bio-inspired Computational Optimization of Speed in an Unplanned Traffic and Comparative Analysis Using Population Knowledge Base Factor", Springer, Advances in Computer Science, Engineering & Applications, 2012, pp. 977-987

[10] P Ghosal, A Chakraborty, S Banerjee, "Speed Optimization in an Unplanned Lane Traffic Using Swarm Intelligence and Population Knowledge Base Oriented Performance Analysis", Springer, Advances in Computer Science, Engineering & Applications, 2012, pp. 471-480